EP1083483A1

Publication Title:

Software migration on an active processing element

Abstract:

Abstract of EP 1083483

(A1) An active processing element undergoes a software migration while under the control of an in-service original software system. A replacement software system is loaded into an available memory partition within the active processing element while the original software system continues to control the active processing element and its services. The original software system continues to control the active processing element as the replacement software system is configured and synchronized with the original software system.; Once a set of dynamic state information within the replacement software system is synchronized to permit the replacement software to take control of at least a portion of the services supported by the processing element, control over such portion of services is transferred from the original software system to the replacement software system in response to a predetermined event such as a command from the system administrator or a fault condition detected in the original software system. As control over services provided by the processing element passes to the replacement software system, corresponding components of the original software system are placed in an inactive state.; Once control over all services provided by the processing element is transferred to the replacement software system, the original software system is rendered fully inactive, although the original software system may continue to use processing resources of the processing element to support the remainder of the software migration, including transferring any remaining state information required by the replacement software system. An image of the original software system is stored on a migration archive for diagnosis.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 083 483 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
14.03.2001 Bulletin 2001/11

(51) Int. Cl.^{7}: **G06F 9/445**

(21) Application number: **00306038.1**

(22) Date of filing: **17.07.2000**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **16.07.1999 US 356044**

(71) Applicant:
Nortel Networks Limited
Montreal, Quebec H2Y 3Y4 (CA)

(72) Inventors:
- **Adamovits, Peter Joseph**
  Ontario, Canada K7C 4E1 (CA)
- **Berube, Louis Pierre**
  Ontario, Canada K0A 1L0 (CA)
- **Dysart, Keith Clifford**
  Ontario, Canada K2S 1E1 (CA)
- **Mussar, Gary P.**
  Ontario, Canada K2H 5E3 (CA)

(74) Representative:
Mackenzie, Andrew Bryan et al
Sommerville & Rushton,
45 Grosvenor Road
St Albans, Herts. AL1 3AW (GB)

(54) **Software migration on an active processing element**

(57) An active processing element undergoes a software migration while under the control of an in-service original software system. A replacement software system is loaded into an available memory partition within the active processing element while the original software system continues to control the active processing element and its services. The original software system continues to control the active processing element as the replacement software system is configured and synchronized with the original software system. Once a set of dynamic state information within the replacement software system is synchronized to permit the replacement software to take control of at least a portion of the services supported by the processing element, control over such portion of services is transferred from the original software system to the replacement software system in response to a predetermined event such as a command from the system administrator or a fault condition detected in the original software system. As control over services provided by the processing element passes to the replacement software system, corresponding components of the original software system are placed in an inactive state. Once control over all services provided by the processing element is transferred to the replacement software system, the original software system is rendered fully inactive, although the original software system may continue to use processing resources of the processing element to support the remainder of the software migration, including transferring any remaining state information required by the replacement software system. An image of the original software system is stored on a migration archive for diagnosis.

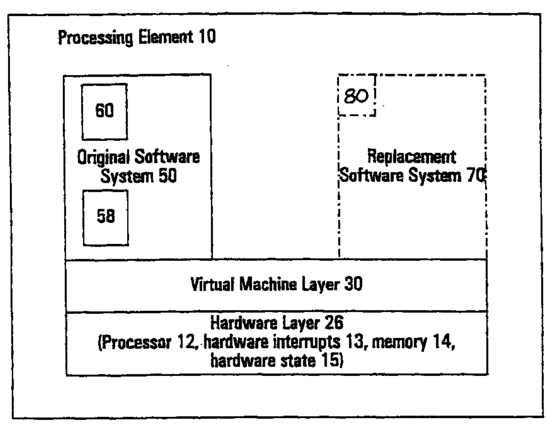


FIG. 2

Printed by Xerox (UK) Business Services
2.16.7 (HRS)/3.6

Description

FIELD

[0001] The present invention relates generally to software migration and, more particularly, to the migration of control over an active processing element from an original software system to a replacement software system.

BACKGROUND

[0002] Performing a software upgrade on computer hardware that is providing services to users or other computer devices presents significant challenges to maintaining the availability of such services during the upgrade. This is particularly the case when the services of the computer hardware are provided at least in part at the direction of the software being replaced in the upgrade. The process of upgrading or replacing the software that controls the computer hardware is often referred to as a software migration and involves replacing the original software with replacement software. During a conventional software migration, the services influenced by the operation of the original software are typically unavailable until the software migration is complete. Such conventional techniques can result in significant periods during which time the computer hardware is unable to provide services controlled by the software being replaced. This problem can become more pronounced in circumstances where there is a need for reliable quality of service, such as in networks and other systems where a high degree of availability may be required for certain services.

[0003] A known solution to such software migration has been to replace the computer hardware with new hardware programmed to provide the new or modified functionality or performance desired. This technique provides a rather crude approach to modifying the services provided by the computer hardware and results in undesirable and extended outages when the computer hardware is unable to support the services it is expected to provide to users, systems or other devices. Such extended outages can be disruptive to both the systems and the users that rely on the operation of the computer hardware. This solution also requires a manual replacement of all or portions of the computer hardware.

[0004] A common approach in the telecommunications industry for reducing the impact of a software migration on the services supported by the computer hardware involves providing duplicate computer hardware. In this latter case, control over available services is passed from the original computer hardware and software to the duplicate computer hardware with replacement software. This approach results in redundant computer hardware sitting idle or under utilized during normal operations when no software migration is taking place, as well as unnecessary complexity in the arrangement and configuration of the original and duplicated computer hardware.

[0005] Software migration may also be necessary after a system failure or crash. In order to restore service after a crash the current software must be overwritten with a replacement software system.

[0006] A problem with this approach is that most systems do not retain the state of the software system that crashed. The memory partition that contained the crashed system is typically overwritten. Fault diagnosis then requires that the conditions that caused the crash be reconstructed. Often, the conditions that existed were not clearly understood, preventing the reconstruction of the state that caused the failure. This prevents the fault from being clearly diagnosed.

[0007] A known solution to this problem is to copy all memory contents to a disk file. This has the cost of time delay. Many software systems now contain hundreds of megabytes resulting in delays of tens of seconds or longer. An alternative is to copy a portion of the crashed software system to disk. This approach has the potential benefit of taking less time but a limitation of this approach is that the failure may have occurred in the portion of memory that was not copied and sufficient information might not be available in the portion copied to conclude the diagnosis.

[0008] Another known solution is to have a duplicate standby hardware and software system to replace the failed system. This approach has a considerably higher cost for the duplicate system plus switching fabrics to switch between the two systems.

[0009] Therefore, it would be desirable to develop an improved mechanism for software migration, which makes more efficient use of installed computer hardware while minimizing the impact on the availability of services during such software migration.

SUMMARY OF THE INVENTION

[0010] In accordance with one aspect of the invention, a method of performing a software migration is provided. In this method, the replacement software system is configured in memory associated with the active processing element and made ready to take control of the active processing element while the in-service original software system controls the active processing element. In order to make the replacement software system ready to take control, state information from the original software system is communicated to the replacement software system.

[0011] Once the replacement software is installed and ready to take control of the active processing element, control over the active processing element is transferred from the original software system to the replacement software system in response to a predetermined event. Such a predetermined event may be, for example, a command by a system administrator or the detection of a fault condition in the original software sys-

tem. Performing a software migration on the active processing element while the original software system controls the active processing element removes the need for having a duplicate processing element to support the migration while substantially maintaining the availability of services on the processing element.

[0012] By simplifying the hardware requirements and localizing much of the software migration to within the domain of the active processing element, the number and severity of service outages on the processing element arising during the software migration are also reduced.

[0013] Although the present invention can be applied to a single processor environment, it also may be applied in a multi-processor (or multi-controller) environment. In fact, when the present invention is applied to a computer system with one or more spare processing elements, processor sparing can be maintained throughout the software migration. In conventional environments, processor sparing or duplication is implemented to provide equipment protection, or in other words, to protect the operation of computer system in the event a running processor, or other computer hardware associated with the processor, fails. In such conventional environments, equipment protection for a processing element is lost during software migration while the spare or duplicate processing element is used in the software migration. Thus, the present invention offers a more robust migration solution that is also applicable to processor spared systems.

[0014] In a preferred embodiment, the replacement software system is configured with provisioning information associated with a hardware and software configuration for the active processing element, and dynamic state information for the replacement software system is synchronized with dynamic state information for the original software system. In the case of the original software system, the dynamic state information includes the state of active services associated with the original software system, as well as hardware state information (of the active processing element) associated with such services. Thus, the dynamic state information for the original software system includes a software representation of the physical hardware state of the active processing element.

[0015] The replacement software system may be loaded in the course of, or prior to, the software migration. Loading and configuring the replacement software system while the original software system continues to run the active processing element avoids losing the services provided by the active processing element during these stages of the software migration. Adding the ability to synchronize the replacement software system with the original software system and the active processing element further reduces the susceptibility of services on the active processing element to extended outages during the migration. Preferably, the configuration and synchronization operations are carried out with the assistance of a virtual machine loaded into available memory within the active processing element and configured to support the transfer of control over the active processing element to the replacement software system. The virtual machine is a software or hardware component (or a combination of hardware and software) which provides a convenient mechanism for facilitating the software migration within the active processing element while minimizing the impact of the software migration on the ability of the active processing element to continue to provide services. The virtual machine may be newly initialized or previously loaded, provided that the virtual machine is configured to support the migration from the in-service original software system to the replacement software system.

[0016] In accordance with another aspect of the invention, an apparatus is provided for supporting a software migration on an active processing element. The apparatus includes a virtual machine and a migration manager. The virtual machine provides an interface during the software migration between the original software system, the replacement software system and hardware elements of the active processing element. In a preferred embodiment, the virtual machine provides several components including a communications path, an interrupt dispatcher, a loader, a system scheduler and a hardware access control module. The communications path facilitates communications between the original software system and the replacement software system during the software migration. The interrupt dispatcher indirectly forwards hardware interrupts from the active processing element to one (or, where appropriate, both) of the original software system and the replacement software system. The loader installs an image of the replacement software system into available memory within the active processing element. The system scheduler schedules at least a portion of the processor resources of the active processing element between the execution of the original software system and the replacement software system during the software migration so that synchronization can take place between the state information for the two software systems while minimizing the impact of the migration on the services provided by the active processing element. The hardware access control module provides an access interface for the original and replacement software systems to access the active processing element. In this way, the hardware access control module protects the active processing element from being corrupted during the software migration.

[0017] According to another aspect of the invention, a system is provided which includes the virtual machine, the migration manager, and the active processing element. In variations where the virtual machine is implemented as a software system, the virtual machine is stored during the software migration in memory within or connected to the active processing element. Once loaded into the active processing element, the virtual

machine serves as an interface between the active processing element and the original and replacement software systems.

[0018] In accordance with yet another aspect of the invention, another software migration system is provided. In this latter system, a unit is included for installing a replacement software system on an active processing element while an original software system controls the active processing element. Another unit is provided for configuring the replacement software system with provisioning information associated with a hardware and software configuration for the active processing element. A unit is also provided with the system in order to synchronize dynamic state information for the replacement software system with dynamic state information for the original software system. An additional unit is provided for transferring control over the active processing element to the replacement software system.

[0019] In another aspect of the invention, a computer readable medium having stored instructions is provided for supporting the installation, configuration and synchronization of a replacement software system on an active processing element while an in-service original software system controls the active processing element. Computer readable codes are also provided for transferring control over the active processing element to the replacement software system.

[0020] In yet another aspect of the invention, an apparatus is provided for supporting a software migration on an active processing element. The apparatus includes a virtual machine implemented to support a transfer of control over the active processing element from an original software system to a replacement software system.

[0021] Other aspects and features of the present invention will become apparent to those ordinarily skilled in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] In the accompanying drawings which illustrate embodiments of the invention,

FIG. **1** is a block diagram of a system for software migration according to a preferred embodiment of the invention;

FIG. **2** is a block diagram of an active processing element of the system for software migration illustrated in FIG. **1**;

FIG. **3** is a block diagram of components of a virtual machine of the active processing element illustrated in FIG. **2**;

FIG. **4** to **12** are block diagrams illustrating stages of the software migration of the active processing element for the preferred embodiment of the invention; and

FIG. **13** is a flow chart showing the execution of the preferred embodiment of invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0023] In this specification, the term "processing element" refers to a processor (or processors) having, or in communication with, sufficient memory to store an in-service original software system, a replacement software system, and additional software supporting the transfer of control from the original software system to the replacement software system. The processor may be, for example, a Power PC, Pentium or the like. When the processing element is "active", it is able to provide services to users or other computer hardware under the direction of a software system. In general, the term "software system" (original, replacement or otherwise) refers to one or more software components programmed to provide one or more services. Such "services" may be any functionality that is useful to the operation of the software system itself or to the operation of other associated computer hardware or software. Services are primarily externally visible to, or support external benefits visible to, users, systems or other computer hardware or other services. "In-service" is used in this specification to mean that the original software system has control over the processing element so as to make available via the processing element the services for which the original software system is programmed to provide. In addition, in the remainder of the specification which follows, the term "software migration" refers to a process of transferring control over the active processing element from the original software system to the replacement software system. As well, the term "service outage" refers here to periods when services, provided by a software system via the active processing element, are unavailable. In this specification, the term "provisioning information" refers to persistent state and configuration information for the software system controlling the active processing element. Provisioning information is meant to survive a power down or reset of the active processing element. The term "dynamic state information" refers to hardware and software state information for the active processing element that varies with time.

[0024] Referring to FIG. **1**, a system for software migration **5**, according to the invention, comprises an active processing element **10**. The active processing element **10** comprises a hardware layer **26** that includes at least one processor **12** connected to memory **14** via a bus **19**. Memory **14** comprises one or more memory banks configured to reserve acceptable amounts of

memory for storing and supporting an in-service original software system **50**, a virtual machine **30** and a replacement software system **70**. Memory **14** may also be configured to provide memory for additional applications, systems and data **25**. For purposes of illustrating the preferred embodiment, memory **14** is a single memory bank having separate memory partitions **18**, **20** and **22** for the original software system **50**, the replacement software system **70** and the virtual machine **30**, respectively.

[0025] The original software system **50** runs and manages the active processing element **10** while the replacement software system **70** is loaded into the second memory partition **20**, provisioning information **48** (including provisioning data and provisioning code) from a provisioning source **49** is transferred to the replacement software system **70**, and dynamic state information **80** for the replacement software system **70** is synchronized with dynamic state information **60** of the original software system **50** (see FIG. **2**). Once at least a minimal set of the dynamic state information **60** is transferred to the replacement software system **70** and transformed and synchronized to the extent sufficient to configure the replacement software system **70** to control at least a portion of the services provided by the active processing element **10**, control over such portion of services passes to the replacement software system **70**. Remaining dynamic state information and any other information required by the replacement software system **70** for full control over the services provided by the active processing element **10** are then transferred from the original software system **50** to the replacement software system **70** and converted to a format recognized by the replacement software system **70**. As control over the services provided by processing element **10** passes to the replacement software system **70**, active components of the original software system **50** which previously provided such services, via the processing element **10**, are retired and rendered inactive.

[0026] Prior to the software migration, the active processing element **10** is loaded with and is under the control of the original software system **50** which directs the services provided by the active processing element **10** to one or more connected clients **17** including, by way of example, another node in a switching network, a disk array, an operator terminal or other computer resources. Such services may be provided over communications fabric **16** or another acceptable communications mechanism. The nature of the services provided by the active processing element **10** under the direction of the in-service original software system **50** will vary with the particular implementation. By way of example, when the active processing element **10** is a functional processor included in a switching system or subsystem, the original software system **50** programs the active processing element **10** to provide and support switching services such as call and connection handling services, Of course, it will be appreciated that other acceptable processing elements (for example, single and multiple CPUs) and other acceptable services may be provided and are considered equivalent.

[0027] During the software migration, the active processing element **10** is loaded with the replacement software system **70** that is installed and configured to take control of the active processing element **10**. Once the replacement software system **70** takes control of the active processing element **10**, the replacement software system **70** directs the active processing element **10** to provide substitute services. Preferably, these substitute services include existing services previously provided by the original software system **50** and additional services not available with the original software system **50**.

[0028] In general terms, the replacement software system **70** may be any vintage of software and any number of software components configured to direct the active processing element **10** to provide one or more services. Thus, the replacement software system **70** may be, by way of example, an earlier version of the original software system **50**, the same version of the original software system **50**, or a subsequent version of the original software system **50** currently active on the active processing element **10**.

[0029] The replacement software system **70** is loaded into the second memory partition **20** (FIG. **1**) with the assistance of the virtual machine **30** which may be either pre-loaded or newly loaded into memory partition **22**. In operation, the virtual machine **30** may be partially or entirely resident in hardware or firmware within the active processing element **10** or it may be loaded directly into memory **14**, although it is preferably fully resident in memory **14** for simplicity, maintenance and substitution.

[0030] Referring to FIG. **2**, the virtual machine **30** provides a migration layer that serves as a virtual interface between hardware layer **26** and software elements for the software migration. The hardware layer **26** comprises the processor **12**, hardware interrupts **13**, memory **14** and hardware state **15**. The virtual machine **30** also provides a migration interface for facilitating the transfer of control of the active processing element **10** from the original software system **50** to the replacement software system **70**. When the original software system **50** or the replacement software system **70** is fully loaded, active and in control of the active processing element **10** under normal operating conditions, the virtual machine **30** is inactive or not present so that as much of the CPU time of the processor **12** as possible is available to support the operations of the software system in control of the active processing element **10**. The original software system **50** comprises memory reserved for a migration manager **58** and dynamic state information for the original software system **50**.

[0031] Referring to FIG. **3**, the virtual machine **30** is programmed to provide several migration-related services which are preferably encapsulated, for simplicity and maintainability, in several software components

including a loader **32**, a communications path **34**, a system scheduler **36**, an interrupt dispatcher **38** and a hardware access control module **40**.

[0032] The remainder of this description depicts the virtual machine **30** as software only. It will be noted, however, that when the virtual machine **30** is implemented in whole or in part as hardware or firmware, the functionality provided by the virtual machine **30** can be substantially the same as a pure software implementation.

[0033] Referring to FIG. **1** to **3**, the loader **32** is configured to load an image of the replacement software system **70** into memory partition **20** and to start the execution of the image. The loader **32** resolves symbolic addresses of the replacement software system **70** at load time so that they are associated with the appropriate physical addresses in the hardware layer **26** and within the software system(s) with which the hardware layer **26** is loaded.

[0034] The communications path **34** provides a mechanism for establishing communication between the original software system **50** and the replacement software system **70** over the active processing element **10**. As well, the communications path **34** provides a basis for communication between each of these systems (**50** and **70**) and, via the virtual machine **30**, the components of the hardware layer **26** and external hardware and software components of other connected resources. The communications path **34** is preferred for single processor embodiments and implementations wherein both the original software system **50** and the replacement software system **70** reside on the same active processing element **10** during the software migration. In this latter case, the communications path **34** provides a safety mechanism so that processes of the original software system **50** do not write directly to memory dedicated to the replacement software system **70** and vice versa. In a spared processor environment, such as with a spared control processor environment, if there is only the original software system **50** or the replacement software system **70** present on a processor at any one time, then the benefits of the communications path **34** are not as significant. In the preferred embodiment, the communications path **34** is preferably a shared or local memory message queue comprising a queue of memory (preferably fixed) which is not part of either memory partition **18** or **20**.

[0035] During the software migration, the system scheduler **36** handles the scheduling of the processor **12** resources and context switching on the processor **12** between the execution of the original software system **50** and the execution of the replacement software system **70**. Preferably, the system scheduler **36** is implemented to provide time-shared scheduling with the ability to release the resources of the processor **12** for use by the waiting software system (original **50** or replacement **70**) if such resources are not required by the currently scheduled software system. In a preferable mode of operation, the allocation of the processor **12** resources between the original software system **50** and the replacement software system **70** via the system scheduler **36** is substantially independent of how the processor **12** is managed by the software system (original or replacement) to which it is allocated at any one time.

[0036] The interrupt dispatcher **38** includes computer-readable code configured to intercept hardware interrupts for the active processing element **10** during the software migration and to forward such hardware interrupts to the appropriate interrupt handler of the software system (original **50** or replacement **70**) in control of the active processing element **10**. By intercepting such hardware interrupts and dispatching them to the appropriate software system (original **50** or replacement **70**) during the software migration, the interrupt dispatcher **38** provides a convenient mechanism for managing hardware interrupts and for ensuring that the hardware interrupts are forwarding to the appropriate software system at each stage of the software migration. In general, prior to control over the active processing element **10** passing to the replacement software system **70**, the hardware interrupts are forwarded by the interrupt dispatcher **38** to the original software system **50**. Once control passes to the replacement software system **70**, the interrupt dispatcher **38** forwards the hardware interrupts to replacement software system **70**.

[0037] In the preferred embodiment, the interrupt dispatcher **38** uses an interrupt table **39** to forward interrupts to the appropriate software system (original **50** or replacement **70**). The interrupt table **39** includes an array of vectors comprising a first and second set of interrupt points addressed to one or the other of the original software system **50** and the replacement software system **70**, or both. One example where certain vectors in the interrupt table **39** point to both the original software system **50** and the replacement software system **70** during the software migration is in the case of references to system clock interrupts. In this latter case, both software systems (**50** and **70**) need to know the state of the system clock interrupts during the software migration. Other cases may also arise depending upon the architecture of the active processing element **10**.

[0038] The hardware access control module **40** manages the access control to the active processing element **10** and serves as an interface between the hardware layer **26** of the active processing element **10** and the original software system **50** and the replacement software system **70**. The hardware access control module **40** protects the hardware layer **26** from being corrupted with invasive read and write instructions during the software migration from the software system (original **50** or replacement **70**) which does not have control over the accessed components of the hardware layer **26**. Thus, the hardware access control module **40** suppresses both software and hardware interrupts for the software system (original **50** or replacement **70**) that

does not presently control the active processing element **10**. In this way, any invasive reads or writes to hardware registers by the software system (original **50** or replacement **70**) not in control of the active processing element **10** are blocked by the virtual machine **30**.

[0039] Preferably, the virtual machine **30** is context independent and minimizes the amount of hardware resource management it performs over the hardware layer **26** so that the virtual machine **30** is easier to remove from the hardware interface path for the active processing element **10** once the software migration is complete. Thus, the virtual machine **30** preferably facilitates communication between the appropriate software system (original **50** or replacement **70**) and the hardware layer **26**, but leaves hardware management for the active processing element **10** to the software system that currently controls the hardware.

[0040] In FIG. **4** to **12**, steps in the operation of the preferred embodiment are identified by numerical labels that are marked as **100** and higher and that are inserted into lines representing the flow of operations. A summary of these steps is provided in the flowchart of FIG. **13** in which the software migration begins at step **100** with the loading and installation of the migration manager **58** into available memory within the original software system **50**. The migration manager **58** is a software application that is responsible for coordinating portions of the software migration. An example of how the migration manager **58** may be used to coordinate and manage the software migration is illustrated in the description below of the preferred embodiment in operation. It should be noted that the migration manager **58** may also be programmed to report, via an external interface, to an administrator involved in the software migration with respect to the status of the software migration. The migration manager **58** may also handle intervention commands issued by the administrator to interrupt or modify the software migration.

[0041] The migration manager **58** is activated at step **102** in response to a predetermined event recognized by the migration manager **58** as a migration initiation signal. Such a signal may be received from a migration initiator **28** such as an automated or manual command signal received by the migration manager **58** from the system administrator via a network resource connected to the active processing element **10**.

[0042] Alternatively, the migration initiator **28** may be the detection of a fault condition in the original software system **50**. Such a fault, for example, may be caused by an endless loop detected by a watchdog timer, a parity error, an attempt to execute an illegal instruction, a memory address violation, a transient hardware error or the like. The fault causes a trap to be executed that passes control to the migration manager **58**.

[0043] In another alternative arrangement, the migration manager **58** already resides in available memory within the domain of the original software system **50** (as shown in FIG. **4**) awaiting instructions to initiate a software migration.

[0044] Once activated, the migration manager **58** determines at step **104** (FIG. **4**) if a version of the virtual machine **30** is present in the active processing element **10**. When a version of the virtual machine **30** is found to be present, the migration manager **58** determines the version of the virtual machine **30** and assesses whether the version will support the current software migration. If a version of the virtual machine **30** is not present or if the version present will not support the current software migration, then the migration manager **58** programs the processor **12** at step **106** to retrieve and load from a data source **31** an appropriate version of the virtual machine **30** that will support the current software migration.

[0045] The migration manager **58** activates the loaded virtual machine **30** at step **108** (FIG. **5**). When the virtual machine **30** is activated it activates the software components within itself to provide an interface for coordinating the operation of, and communication between, the active processing element **10**, the original software system **50** and the replacement software system **70** during the software migration. Thus, in the preferred embodiment, the loader **32**, the communications path **34**, the system scheduler **36**, the interrupt dispatcher **38** and hardware access control module **40** are each preferably activated at step **108** to support the virtual machine **30**. The communications path **34** is activated in order to facilitate communications between the original software system **50** and the replacement software system **70**. The system scheduler **36** is activated to take control of scheduling of the resources of processor **12** between the original software system **50** and the replacement software system **70** during the software migration. The interrupt dispatcher **38** is activated to redirect hardware interrupts **13** from hardware layer **26** to the original software system **50** or the replacement software system **70** (or both) via an interrupt table **39**. The redirection of hardware interrupts **13** is illustrated by interrupt paths **46** and **47**. Prior to control of the processor **12** passing to the replacement software system **70**, the interrupt dispatcher **38** redirects hardware interrupts **13** to the original software system **50**. The hardware access control module **40** is activated as part of the virtual machine **30** to ensure that hardware registers in the active processing element **10** have read and write protection.

[0046] In operation, the hardware access control module **40** intercepts read and write instructions to hardware components of the active processing element **10** initiated by either the original software system **50** or the replacement software system **70**. Read and write instructions which are intercepted by the hardware access control module **40** are either forwarded to be carried out on the associated hardware component(s) of the active processing element **10** or are blocked depending on the stage of the software migration and

the source of the read and write instructions. Thus, prior to control over any of the services provided by the active processing element **10** passing to the replacement software system **70**, read and write instructions affecting the hardware state **15** of the active processing element **10** and originating from the replacement software system **70** are blocked to protect the integrity of the active processing element **10**. On the other hand, read and write instructions directed to the hardware registers of the processing element **10** from components of the virtual machine **30** itself, such as from the communications path **34** or the system scheduler **36**, are directed to their respective hardware registers in support of the software migration.

[0047] Once the virtual machine **30** is loaded, installed and activated in active processing element **10**, the loader **32** proceeds to retrieve and load an image of the replacement software system **70** into the second memory partition **20** at steps **110** and **112**. These latter steps may be performed under the coordination of the migration manager **58**. The replacement software system **70** is loaded by the virtual machine **30** from a migration source **68** connected, directly or indirectly, to the active processing element **10**. Such a migration source **68** may be, by way of example, flash memory, a network resource, or a permanent or removable storage device such as a hard disk drive, a floppy disk, DVD, a compact disc or the like.

[0048] Preferably, the software migration is performed during a period when the usage of the active processing element **10** is low and resource management of processor **12** is carried out primarily through the system scheduler **36**. Alternatively, when the system scheduler **36** is activated, the replacement software system **70** can be initialized and scheduled to share the resources of the processor **12** with the original software system **50**, although control over the active processing element **10** and the processor **12** remains at this stage with the original software system **50**. Thus, as the replacement software system **70** is initialized and run on the active processing element **10** in this alternative mode, performance levels for the services provided by the original software system **50** are reduced, or "throttled-down", to allow the resources of processor **12** to be shared between the original software system **50** and the replacement software system **70**. Such throttling of the services for the original software system **50** may be enforced by requiring a minimum elapsed time between the initiation of available services.

[0049] With the replacement software system **70** running on the active processing element **10**, the migration manager **58** coordinates the transfer of the provisioning information **48** into the replacement software system **70** at step **116** (FIG. **6**). The provisioning information **48** represents the persistent state and configuration information for the software system (original or replacement) controlling the active processing element **10** and which is meant to survive a power down or reset of the active processing element **10**. In the preferred embodiment, the provisioning information **48** is loaded indirectly into the replacement software system **70** from a provisioning source **49** via the original software system **50**. This is achieved by time-slicing with the system scheduler **36** between the original and replacement software systems (**50** and **70**) and by using the communications path **34** to transfer the provisioning information **48** to the replacement software system **70** via the original software system **50**. In an alternative mode of operation, the provisioning information may be transferred directly from the provisioning source **49** to the replacement software system **70** via the virtual machine **30**. Such a provisioning source may include a hard disk, flash memory, another processing element, or another resource having an acceptable storage device or memory device. The provisioning source **49** may even be a memory partition within memory **14** of the active processing element **10**.

[0050] It is worth noting that the replacement software system **70** can take many forms and may comprise a single compiled set of computer-readable code or multiple compiled images which are loaded as required (see FIG. **6**). Furthermore, loading the replacement software system **70** into the second memory partition **20** may proceed in several stages. By way of example, in the preferred embodiment the replacement software system **70** comprises a multi-image system, including a kernel **72**, base code **74** and one or more applications **76**, which are loaded in stages. The kernel **72** provides basic I/O operations for supporting communication between higher layer components of the replacement software system **70** and the hardware components of the active processing element **10**. The base code **74** represents components of the replacement software system **70**, including, by way of example, an operating system such as UNIX, VMX, QNX or the like, which are required to support the operation of several or all of the higher level applications **76** which are configured to provide some or all of the services to be provided by the active processing element **10** once it is under the control of the replacement software system **70**.

[0051] In the preferred embodiment, the kernel **72** is loaded into the second memory partition **20** at the direction of the virtual machine **30** including the loader **32**. Preferably, the loader **32** loads the kernel **72** into memory **14** using an absolute base address defined for use by the kernel **72**, although in a variation on the preferred embodiment, the kernel **72** may be relocatable to other addresses. Additional components of the replacement software system **70** are loaded into acceptable memory locations within memory **14** relative to the base address defined for the kernel **72** and have their symbolic addresses resolved at load time based on the base address definition.

[0052] Once the kernel **72** is loaded into the second memory partition **20**, the kernel **72** is activated and the

system scheduler **36** proceeds to share at least a portion of the processing resources of the processor **12** between the kernel **72** and the original software system **50**. Further components of the replacement software system **70** are then loaded and initialized with the assistance of the kernel **72**.

[0053] Preferably, the components loaded as part of the replacement software system **70** are demand driven so as to make efficient use of memory **14**. Thus, when the base code **74** is loaded by the kernel **72**, the type of active processing element **10** is identified by the kernel **72** so as to determine what type of provisioning information **48** is necessary to configure components of the base code **74** such as the operating system for the particular hardware configuration of the active processing element **10** in use. Thus, configuring (or initializing) the replacement software system **70** with the necessary provisioning information **48** in the preferred embodiment is preferably an interactive operation that cooperates with the loading of the replacement software system **70** early in the software migration. Configuration of the replacement software system **70** preferably begins even before all of the code for the replacement software system **70** is loaded. In fact, the initial components of the provisioning information **48** which are used to configure the base code **74** may also contribute to determining what software code and data will be loaded into the second memory partition **20** as part of the applications **76** for the replacement software system **70**. Once the replacement software system **70** is loaded, more specific initialization operations are carried out by the base code **74** to configure specific services provided by the active processing element **10** that are to be operational when the migration to the replacement software system **70** is complete.

[0054] After the replacement software system **70** has been configured with the provisioning information **48** (see step **116** of FIG. **6**), the migration manager **58** initiates the transfer of dynamic state information **60** of the original software system **50** to the replacement software system **70** via the communications path **34** at step **120**. In general, dynamic state information represents hardware and software state information for the active processing element **10** which vanes with time. In the case of the original software system **50**, the dynamic state information **60** includes the state of active services associated with the original software system **50**, as well as hardware state **15** information (of the active processing element **10**) associated with such services. Thus, the dynamic state information **60** for the original software system **50** includes a software representation of the physical hardware state **15** of the active processing element **10**.

[0055] The dynamic state information **60** is preferably transferred to the replacement software system **70** via I/O interfaces **62** and **82** and communications path **34** (FIG. **7**). The I/O interfaces **62** and **82** provide a synchronization mechanism for transferring the dynamic state information **60** to the replacement software system **70** for transformation into the dynamic state information **80**, and for synchronizing the dynamic state information **80** of the replacement software system **70** with the dynamic state information **60** of the original software system **50** at least until control over the active processing element **10** passes to the replacement software system **70**. Preferably, the dynamic state information **60** represents dynamic state information that is critical to synchronizing state information for the replacement software system **70** with state information for the original software system **50**. Thus, portions of the dynamic state information of the original software system **50** which are inconsequential to the transfer of control over the active processing element **10** or to the synchronization of dynamic state information between the original software system **50** and the replacement software system **70** for the purposes of maintaining services may be discarded during or following the transfer of control.

[0056] As dynamic state information **60** from the original software system **50** is received by the I/O interface **82** of the replacement software system **70**, the dynamic state information **60** is transformed into a format recognized by the replacement software system **70**. This transformation includes mapping the dynamic state information **60** into data structures recognized by the replacement software system **70**. The transformed dynamic state information **60** is stored as the dynamic state information **80** for the replacement software system **70**. Once the replacement software system **70** is loaded with the dynamic state information **80**, the migration manager **58** coordinates the synchronization of the dynamic state information **80** with the dynamic state information **60** of the original software system **50** (as illustrated at step **121** in FIG. **7**) until control over the active processing element **10** passes to the replacement software system **70**. Control over the active processing element **10** is transferred to the replacement software system **70** as existing services provided by the original software system **50** are replaced by services provided by the replacement software system **70**. Thus, the synchronization between the dynamic state information **80** of the replacement software system **70** and its original counterpart **60** is maintained by the synchronization mechanism provided by I/O interfaces **62** and **82** at least until the existing services controlling the active processing element **10** are replaced by the services of the replacement software system **70** (as illustrated by step **123** in FIG. **8**). In order to maintain such a synchronization, state changes to the dynamic state information **60** for the original software system **50** are recorded as such state changes occur. The recorded information is then transferred to the replacement software system **70** where it is used to synchronize the dynamic state information **80**.

[0057] Preferably, in order to minimize the degree to which active services provided by the active processing

element **10** are impacted, the dynamic state information **60** transferred and transformed into dynamic state information **80** at step **120** is sufficient to allow the replacement software system **70** to take control of a subset of the services supported by the active processing element **10**. For example, at this stage an image of the dynamic state information **60** sufficient to permit the replacement software system **70** to control new requests for services may be transferred, transformed and synchronized while the remaining dynamic state information **60** is temporarily maintained within the domain of the original software system **50**. In another alternative mode of operation, the dynamic state information **60** that is transferred and transformed into the dynamic state information **80** is sufficient to permit the replacement software system **70** to take control of all services supported by the active processing element **10**.

[0058] Once the transferred subset of dynamic state information is transformed into part of the dynamic state information **80** of the replacement software system **70** and synchronized with the corresponding dynamic state information **60** in the original software system **50**, the migration manager **58** activates at step **122** (FIG. **8**) the replacement software system **70** and passes control of the active processing element **10** from the original software system **50** to the replacement software system **70** to the extent such passing of control is supported by the transferred and transformed portion of the dynamic state information **60**. In conjunction with step **122**, the migration manager **58** invokes the virtual machine **30** at step **124** to instruct the replacement migration manager **84**, loaded as part of the replacement software system **70**, to take control of the software migration. For the preferred embodiment illustrated, this latter instruction is communicated to the replacement migration manager via the communication path **34**.

[0059] Once the replacement software system **70** has been activated, all new requests for services supported by the dynamic state information **80** that require handling by the active processing element **10** are processed by the replacement software system **70**, with the original software system **50** being blocked by the replacement migration manager **84** from handling such new requests. All control messages for controlling the active processing element **10**, at least in respect of the services activated by the new requests, are also handled by the replacement software system **70** following its activation.

[0060] At this stage, the original software system **50** is quiesced at step **126** at the direction of the replacement migration manager **84** via the communications path **34** (FIG. **9**). Quiescing the original software system **50** involves placing it into a stable inactive state wherein it no longer controls hardware elements of the active processing element **10**, although the original software system **50** may still use processing resources of the processor **12** to handle data for any remaining synchronization between the original software system **50** and the replacement software system **70**. If control over the ability of the active processing element **10** to provide all available services is passed to the replacement software system **70** in a single stage, then the original software system **50** is fully quiesced. However, if control passes in several stages, then the original software system **50** is only partially quiesced a this stage and the corresponding components of the original software system **50** required to continue to support services not yet under the control of the replacement software system **70** remain active.

[0061] During step **126**, there may be a temporary, yet complete suspension of services provided by the active processing element **10**. This temporary, complete loss of service is preferably limited to no more than the period in which control over the active processing element **10** actually passes in step **126** from the original software system **50** to the replacement software system **70**. In this way, the outage period in which the active processing element **10** is unable to provide any services is minimized to a small portion of the software migration.

[0062] Following the passing of at least partial control over the active processing element **10** in step **126** to the replacement software system **70**, services which were temporarily suspended are resumed. In the case where performance levels of services were previously throttled-down during the software migration, the processing of supported services is still throttled down below maximum allowable processing resources to allow the processor **12** to run remaining components of the original software system **50** and to support the remainder of the software migration.

[0063] Active services existing prior to control of the active processing element **10** passing to the replacement software system **70** preferably continue to be handled by the original software system **50** until those active services are transferred to the replacement software system **70** under the coordination of the replacement migration manager **84** in step **128** (FIG. **10**) or, alternatively, under the coordination of the migration manager **58**. This preferable mode of operation provides a convenient mechanism for passing primary control of the active processing element **10** over to the replacement software system **70** without having to simultaneously transfer control of all previously existing services handled by the active processing element **10**. In the alternative mode, control over previously existing and new services may pass simultaneously to the replacement software system **70**, although this will result in a more significant instantaneous impact on the ability of the active processing element **10** to maintain services as greater processing resources are required to perform such a simultaneous task.

[0064] Once control over all services provided by the processing element **10** are transferred (for instance, at step **128** in the preferred embodiment), the replacement software system **70** takes complete control over

the processing element **10**. When control over the processing element **10** passes to the replacement software system **70**, the hardware interrupts **13** on the processing element **10** are forwarded to the replacement software system **70** through the interrupt table **39** managed by the interrupt dispatcher **38**. Redirecting the hardware interrupts **13** may occur in stages (as shown, by way of example, in FIG. **9** to **11**).

[0065] Advantageously, up to step **124**, the software migration can be reversed without requiring significantly more overhead or steps. Before the replacement software system **70** is activated, rolling back the software migration merely involves releasing the newly allocated resources for the replacement software system **70** and unloading the replacement software from the second memory partition **20**. After new services have been initiated with the replacement software system **70** or previous services have been transferred to the replacement software system **70** for further operation, a migration rollback involves transferring such new or previous services and the corresponding new or modified dynamic state information back to the domain of the original software system **70**.

[0066] In an alternative embodiment in which a migration rollback is supported following step **124**, the synchronization of dynamic state information **60** and **80** is supported in the opposite direction, wherein data having the form recognized by the replacement software system **70** is converted into a form suitable for use by the original software system **50**. In this latter configuration, the virtual machine **30** and the original and replacement migration managers (**58** and **84**) are configured to support bidirectional migration of provisioned information and dynamic state information. Performing the software migration in reverse, from the replacement software system **70** to the original software system **50**, is substantially the same as the opposite direction with the exception that the translation of dynamic state information is preferably performed in the replacement software system **70** before being transferred to the original software system **50**. The translation of dynamic state information occurs, in this case, in the replacement software system **70** since the original software system **50** will have no possible knowledge of subsequent data structures found in the replacement software system **70**.

[0067] Following the activation of the replacement software system **70**, remaining elements of the dynamic state information **60** which were not previously transferred are passed to the replacement software system **70** at step **130** and synchronized under the coordination of the replacement migration manager **84** and the virtual machine **30** (FIG. **11**). With all control of the active processing element **10** having been passed to the replacement software system **70** and all dynamic data transfer and synchronization complete, the replacement migration manager **84** fully disables the original software system **50** at step **132** and instructs the virtual machine **30** to schedule all processing resources of the processor **12** for the replacement software system **70** at step **134** (FIG. **11**). The replacement migration manager **84** then instructs the virtual machine **30** at step **136** to remove all references to software elements which were part of the execution of the original software system **50**, including the deregistration of functions and any objects, and the deletion of tasks and components which were part of the original software system **50**. At step **136**, the replacement migration manager **84** further coordinates the release of all system resources within the active processing element **10** which were allocated by the original software system **50**, including heap memory, semaphores, sockets, file handles and the like (FIG. **11**). Next, in step **138**, the original software system **50** is unloaded by the virtual machine **30** and the first memory partition **18** is returned to the system resource pool for the active processing element **10**.

[0068] In an alternative mode, an image of original software system **50** is stored on a migration archive **61** following step **138**. Such a migration archive **61** may be, for example, flash memory, a network resource, or a permanent or removable storage device such as a hard disk drive, a floppy disk, a writable DVD, a writable compact disc or the like. The storing of an image of the original software system **50** on a migration archive **61** is particularly useful for post-migration diagnosis after the detection of a fault in the original software system **50**.

[0069] In another alternative mode, the pages of memory in the first memory partition **18** are reserved for future migrations.

[0070] With all processing resources of the processor **12** allocated to the replacement software system **70**, the throttling-down of processing services, when included in an implementation, is preferably turned off at step **136** by the migration manager **58** (FIG. **11**). Once full control over the active processing element **10** has passed to the replacement software system **70** and the original software system **50** has been completely disabled, the migration services provided by the virtual machine **30** are no longer required by the active processing element **10** (see FIG. **11**). At this stage, the components of the virtual machine **30** are preferably disabled, including the loader **32**, the communications path **34**, the system scheduler **36**, the interrupt dispatcher **38** and the hardware access control module **40**. Disabling the virtual machine **30** upon completion of the software migration minimizes the parasitic load on the processor **12**. With the virtual machine **30** disabled, the replacement software system **70** interfaces directly with the hardware layer of the active processing element **10** and interrupts are forwarded directly to the replacement software system **70**, as illustrated in FIG. **12**, rather than indirectly through the interrupt dispatcher **38**, as illustrated, for example, in FIG. **11**. Although it is disabled, the virtual machine **30** may continue to reside, in whole or in part, in memory **14**, or it may be removed altogether.

[0071] Although this invention has been described with reference to illustrative and preferred embodiments of carrying out the invention, this description is not to be construed in a limiting sense. Various modifications of form, arrangement of parts, steps, details and order of operations of the embodiments illustrated, as well as other embodiments of the invention, will be apparent to persons skilled in the art upon reference to this description. It is therefore contemplated that the appended claims will cover such modifications and embodiments as fall within the true scope of the invention.

Claims

1. A method of performing a software migration, the method comprising:

   (a) configuring a replacement software system in memory associated with an active processing element while an original software system controls the active processing element;

   (b) communicating state information from the original software system to the replacement software system so as to prepare the replacement software system to take control of the active processing element; and

   (c) transferring control of the active processing element to the replacement software system in response to a predetermined event.

2. The method of claim **1**, wherein the predetermined event is a command from a system administrator.

3. The method of claim **1**, wherein the predetermined event is a fault condition detected in the original software system.

4. The method of claim **3**, further comprising storing an image of the original software system in a migration archive.

5. The method of claim **1**, wherein communicating comprises synchronizing dynamic state information for the replacement software system with dynamic state information for the original software system.

6. The method of claim **4**, further comprising maintaining the synchronization of the dynamic state information for the replacement software system with the dynamic state information for the original software system at least until services provided by the original software system via the active processing element are replaced by services provided by the replacement software system.

7. The method of claim **1**, further comprising disabling the original software system once control over the active processing element passes to the replacement software system.

8. Apparatus for supporting a software migration on an active processing element from an original software system to a replacement software system, the apparatus comprising:

   (a) a virtual machine for interfacing between the active processing element, the original software system and the replacement software system; and

   (b) a migration manager for coordinating, in cooperation with the virtual machine, a transfer of control over the active processing element from the original software system to the replacement software system in response to a predetermined event;

9. The apparatus of claim **8**, wherein the predetermined event is a command from the system administrator.

10. The apparatus of claim **8**, wherein the predetermined event is a fault condition detected in the original software system.

11. The apparatus of claim **10**, further comprising a migration archive for storing an image of the original software system.

12. A system comprising the apparatus of claim **8**, the system further comprising:

    (a) an active processing element; and

    (b) an interface for synchronizing and maintaining synchronization, in cooperation with the virtual machine, dynamic state information of the replacement software with dynamic state information of the original software system during the software migration on the active processing element.

13. The system of claim **12**, wherein the predetermined event is a command from the system administrator.

14. The system of claim **12**, wherein the predetermined event is a fault condition detected in the original software system.

15. The system of claim **14**, further comprising means for storing an image of the original software system in a migration archive.

16. The system of claim **12**, further comprising means

for disabling the original software system once control over the active processing element passes to the replacement software system.

17. A computer readable medium having stored instructions comprising:

(a) computer readable codes for installing a replacement software system on an active processing element while an original software system controls the active processing element;

(b) computer readable codes for configuring the replacement software system with provisioning information associated with a hardware and software configuration for the active processing element;

(c) computer readable codes for synchronization and maintaining synchronization of dynamic state information for the replacement software system with dynamic state information for the original software system; and

(d) computer readable codes for transferring control over the active processing element to the replacement software system in response to a predetermined event.

18. The computer readable medium of claim **17**, wherein the predetermined event is a command from the system administrator.

19. The computer readable medium of claim **17**, wherein the predetermined event is a fault condition detected in the original software system.

20. The computer readable medium of claim **19**, further comprising computer readable codes for storing an image of the original software system in a migration archive.

21. The computer readable medium of claim **17**, further comprising computer readable codes for disabling the original software system once control over the active processing element passes to the replacement software system.

22. A system for performing a software migration, comprising:

(a) means for installing a replacement software system on an active processing element while an original software system controls the active processing element;

(b) means for configuring the replacement software system with provisioning information associated with a hardware and software configuration for the active processing element;

(c) means for synchronizing dynamic state information for the replacement software system with dynamic state information for the original software system; and

(d) means for transferring control over the active processing element to the replacement software system.

23. The system of claim **22**, further comprising means for managing the transfer of control over the active processing element to the replacement software system during the software migration.

24. The system of claim **23**, further comprising means for disabling the original software system once control over the active processing element passes to the replacement software system.

25. The system of claim **24**, further comprising means for communicating between the replacement software system and the original software system during the software migration.

26. The system of claim **25**, further comprising means for allocating processor resources of the active processing element between the original software system and the replacement software system independent of how the processor resources are managed within either of the original software system and the replacement software system.

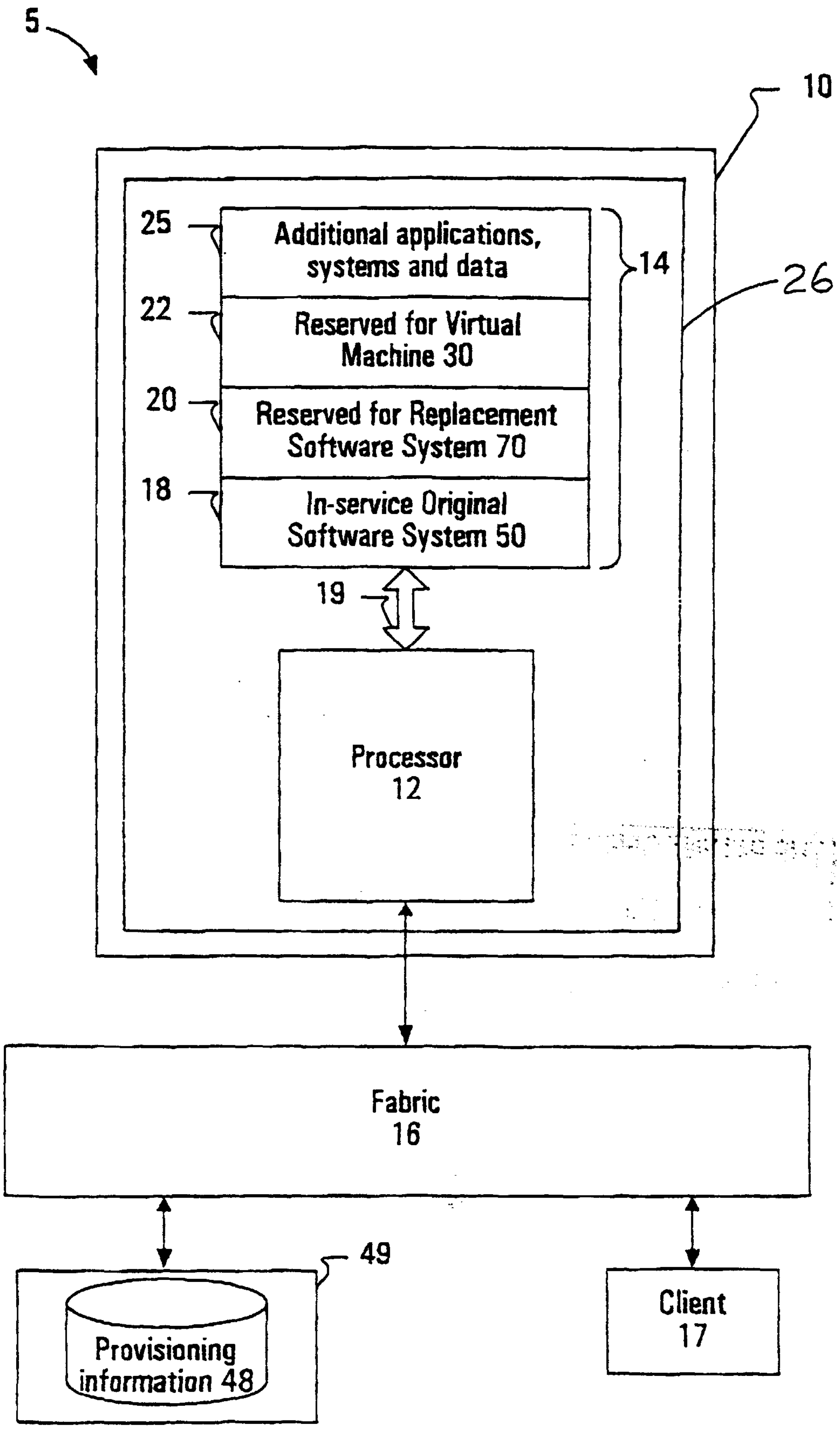

5
10
25
Additional applications, systems and data
22
Reserved for Virtual Machine 30
20
Reserved for Replacement Software System 70
18
In-service Original Software System 50
14
26
19
Processor 12
Fabric 16
49
Provisioning information 48
Client 17

FIG. 1

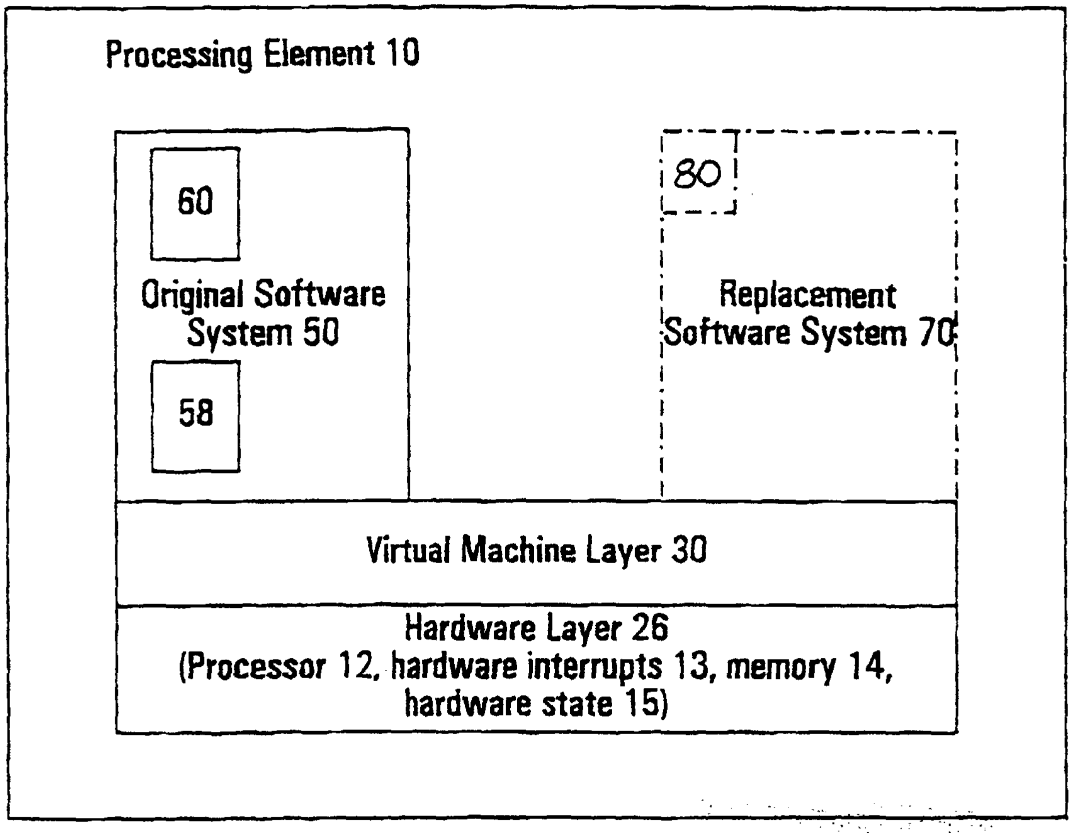

Processing Element 10
60
Original Software
System 50
58
80
Replacement
Software System 70
Virtual Machine Layer 30
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14,
hardware state 15)

FIG. 2

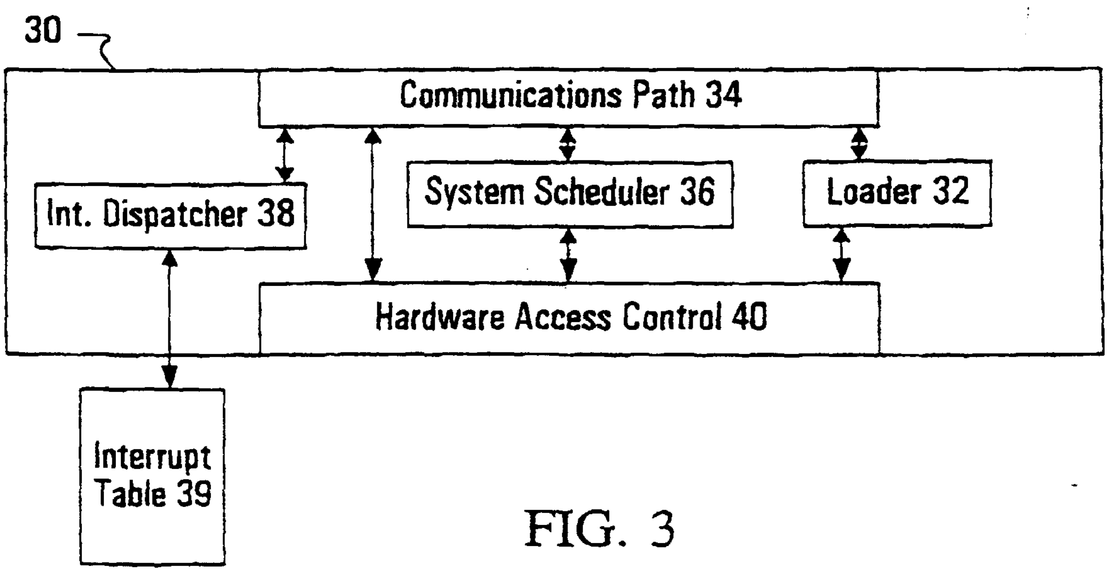

30
Communications Path 34
Int. Dispatcher 38
System Scheduler 36
Loader 32
Hardware Access Control 40
Interrupt
Table 39

FIG. 3

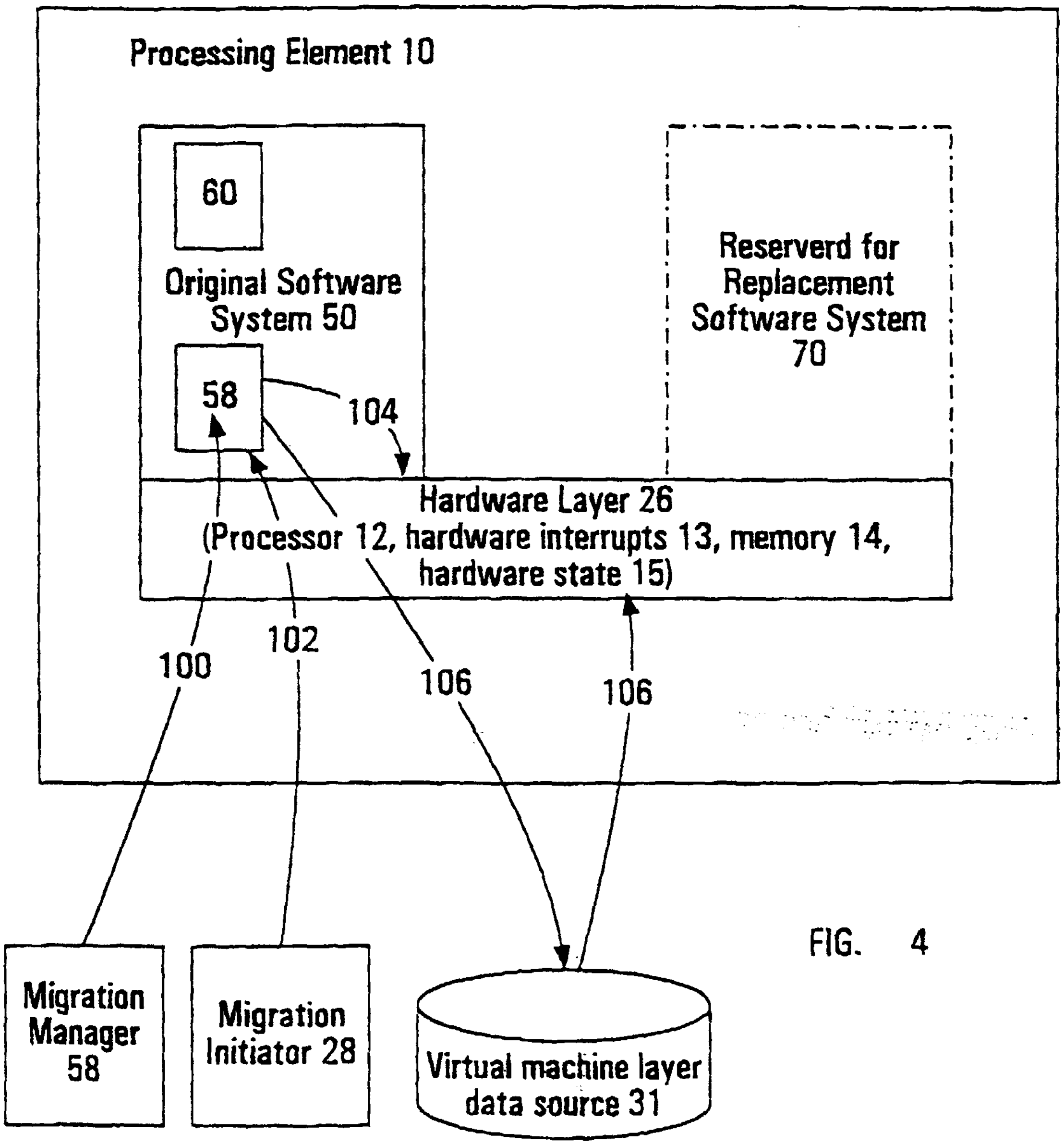

Processing Element 10
60
Original Software System 50
58
104
Reserverd for Replacement Software System 70
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14, hardware state 15)
100
102
106
106
Migration Manager 58
Migration Initiator 28
Virtual machine layer data source 31

FIG. 4

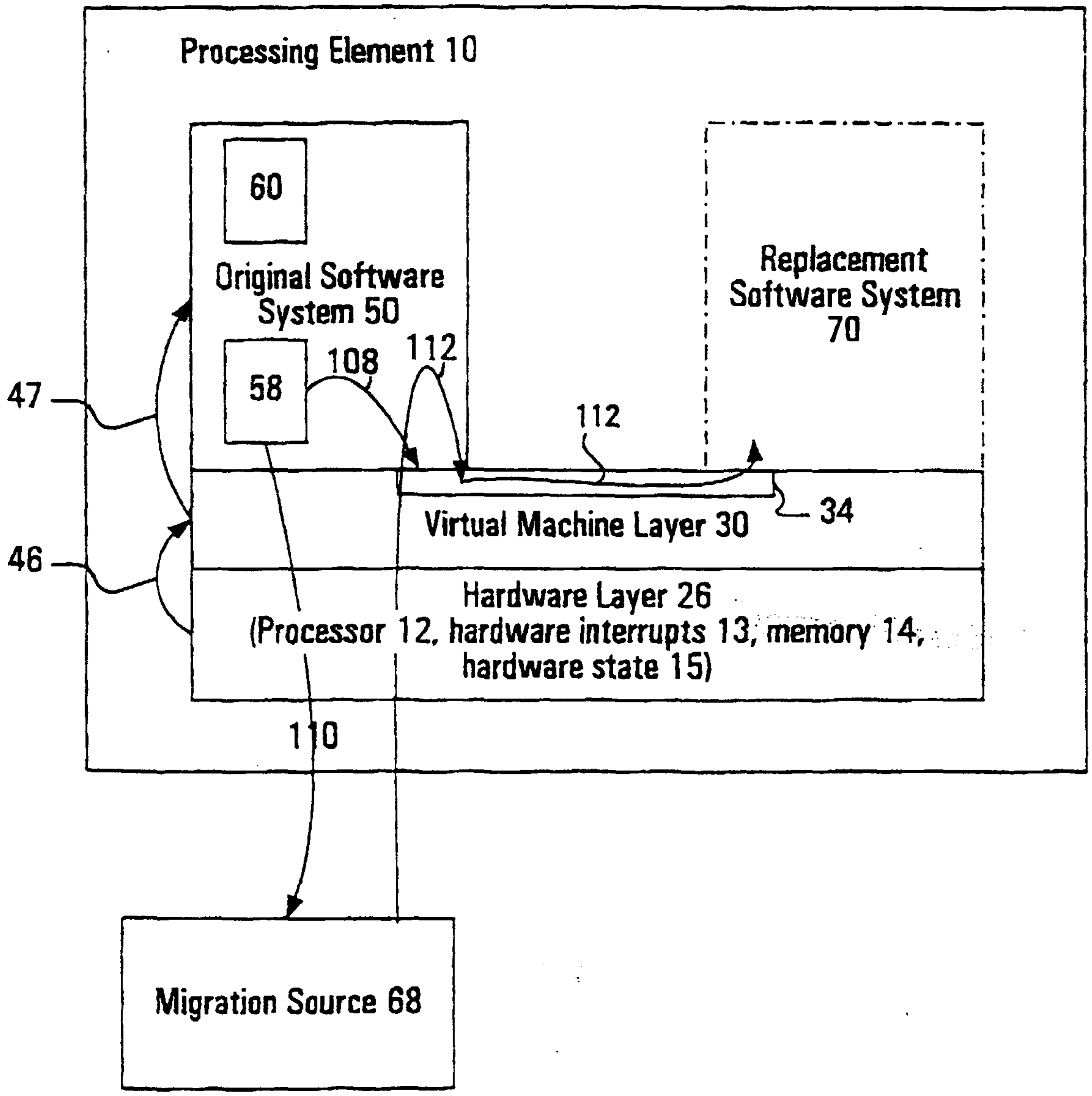

Processing Element 10
60
Original Software System 50
58
108
112
112
Replacement Software System 70
47
46
34
Virtual Machine Layer 30
Hardware Layer 26 (Processor 12, hardware interrupts 13, memory 14, hardware state 15)
110
Migration Source 68

FIG. 5

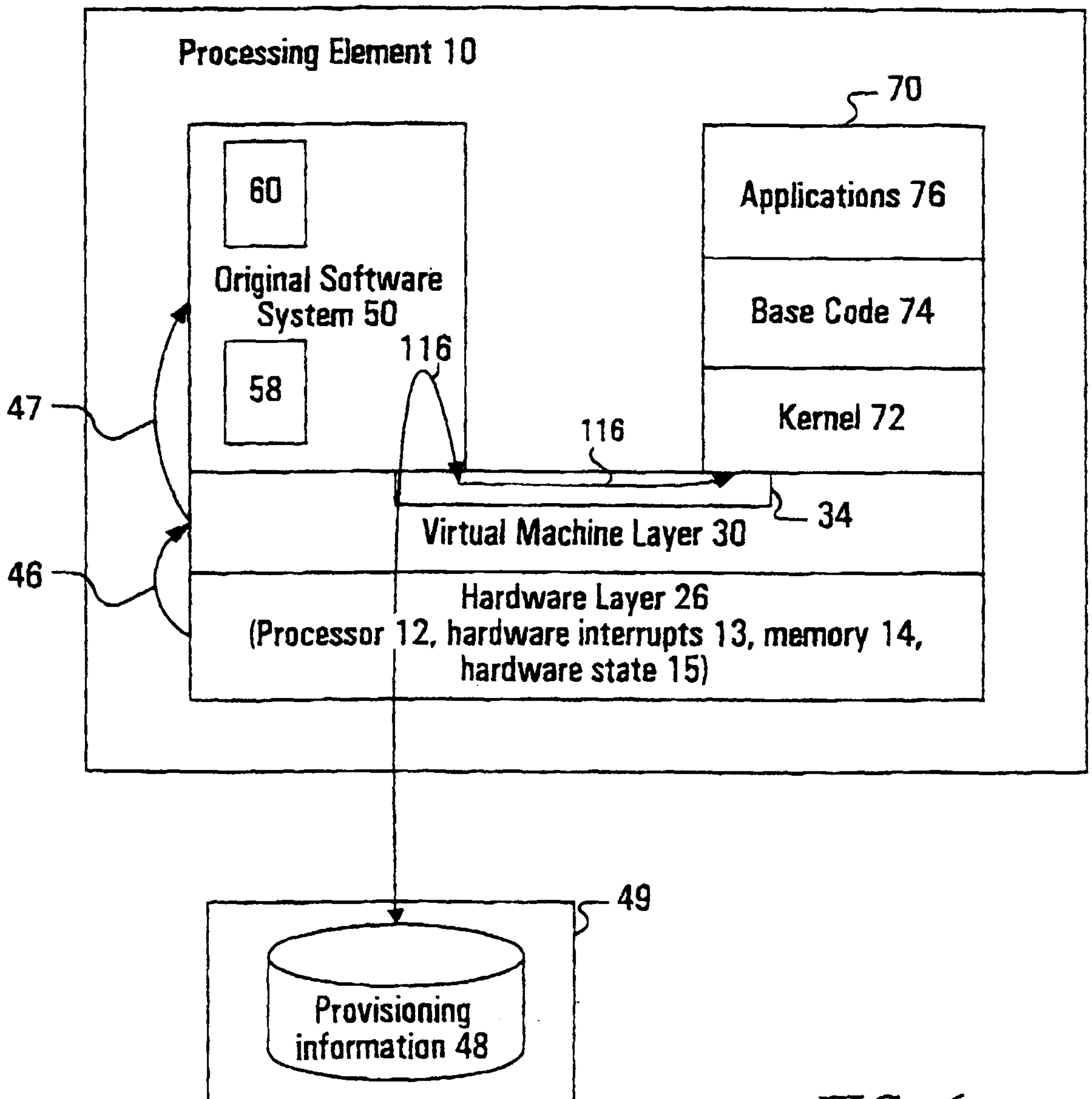

Processing Element 10
70
60
Applications 76
Original Software System 50
Base Code 74
116
58
47
116
Kernel 72
34
Virtual Machine Layer 30
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14, hardware state 15)
49
Provisioning information 48

FIG. 6

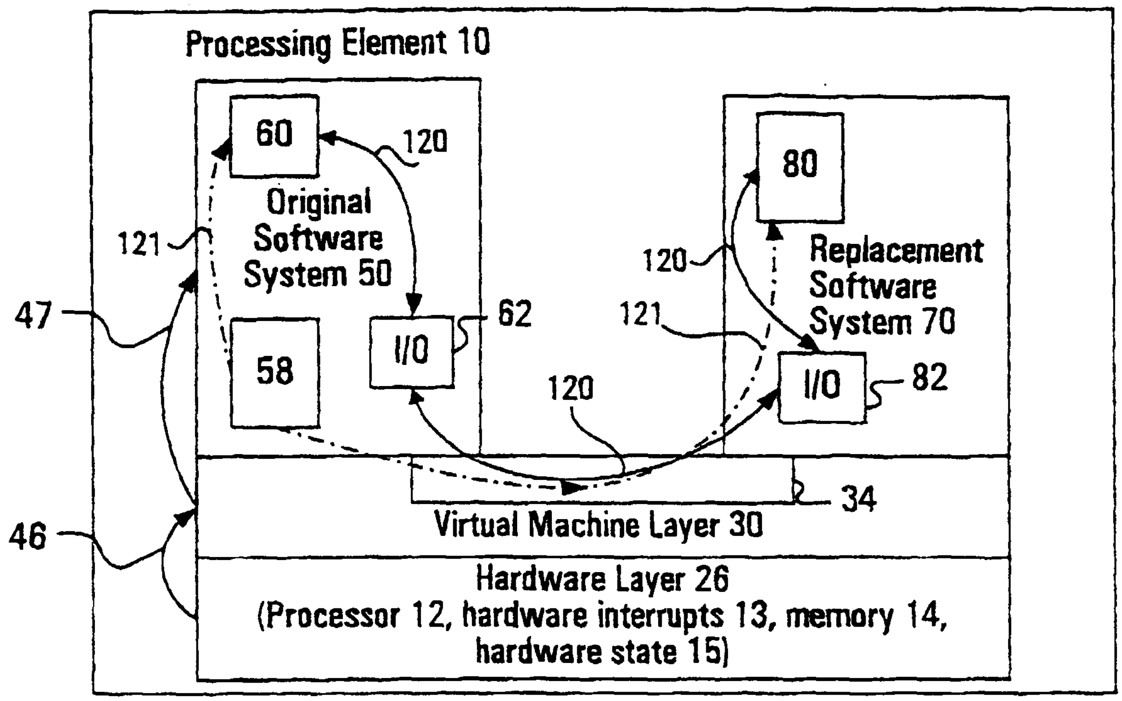

Processing Element 10
60
120
Original
Software
System 50
121
47
58
I/O
62
80
120
Replacement
Software
System 70
121
I/O
82
120
34
Virtual Machine Layer 30
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14,
hardware state 15)

FIG. 7

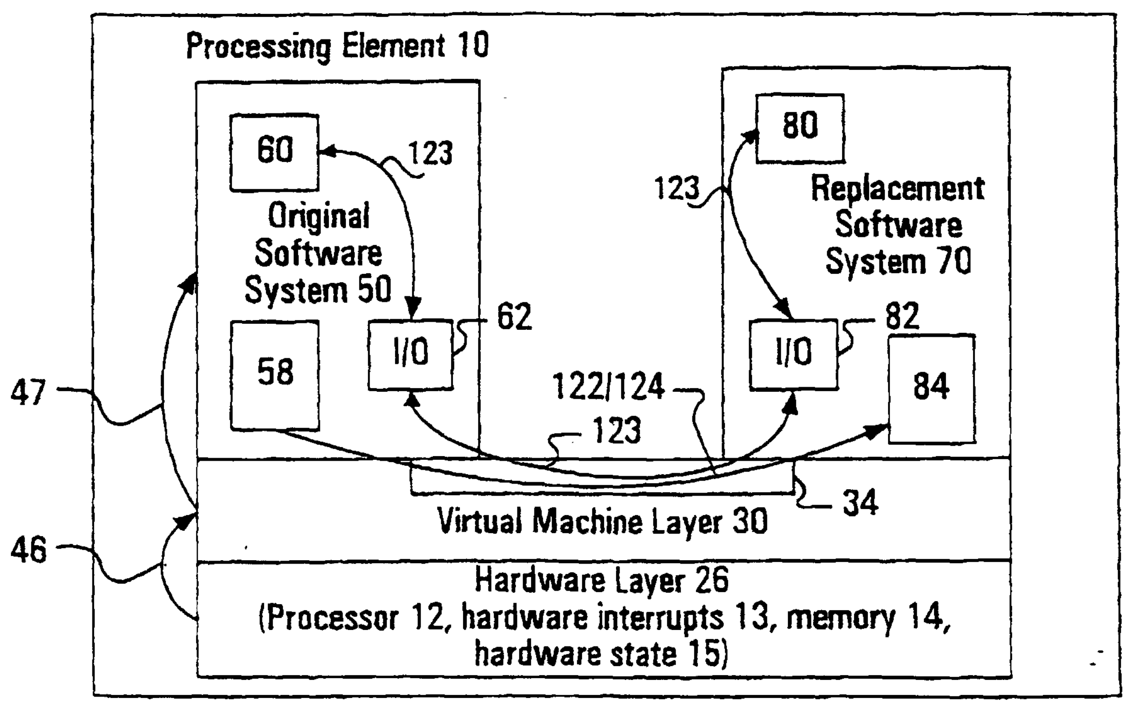

Processing Element 10
60
123
Original
Software
System 50
58
I/O
62
47
80
123
Replacement
Software
System 70
I/O
82
84
122/124
123
34
Virtual Machine Layer 30
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14,
hardware state 15)

FIG. 8

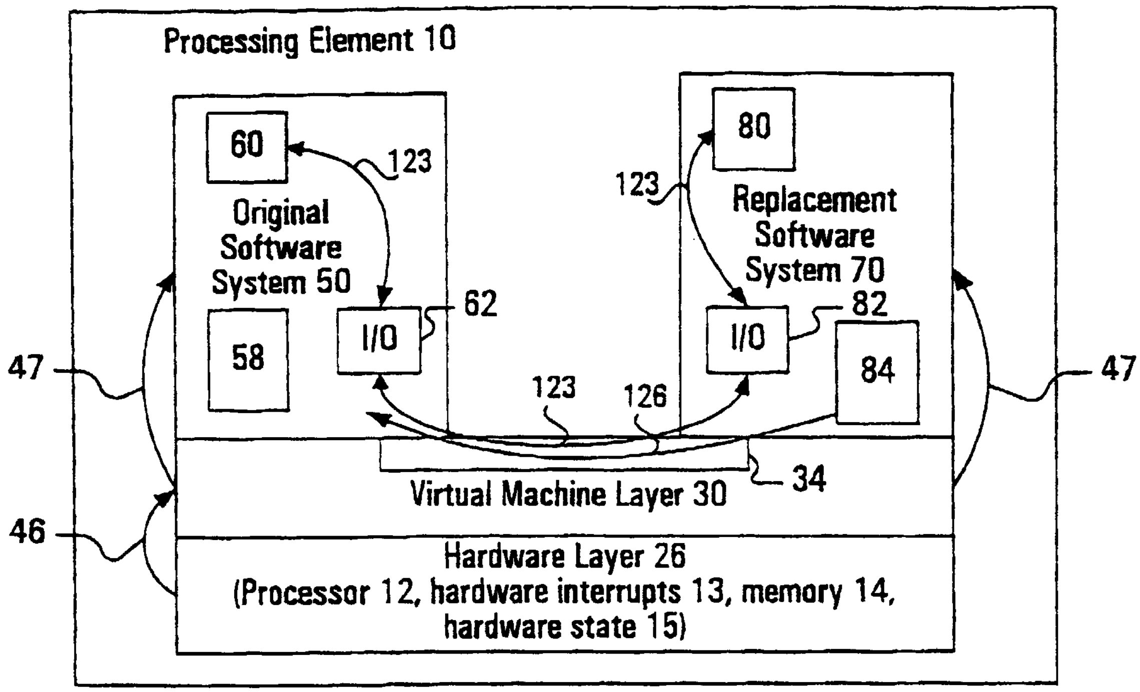

Processing Element 10
60
123
Original
Software
System 50
58
I/O
62
80
123
Replacement
Software
System 70
I/O
82
84
47
47
123
126
34
Virtual Machine Layer 30
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14,
hardware state 15)

FIG. 9

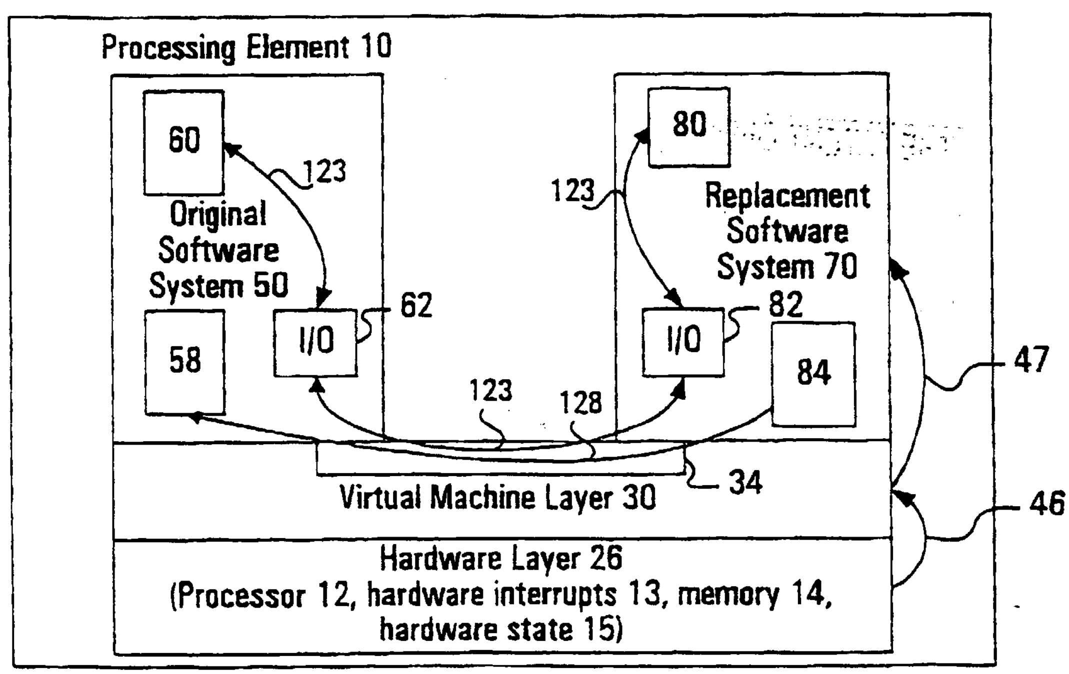

Processing Element 10
60
123
Original
Software
System 50
58
I/O
62
80
123
Replacement
Software
System 70
I/O
82
84
47
123
128
34
Virtual Machine Layer 30
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14,
hardware state 15)

FIG. 10

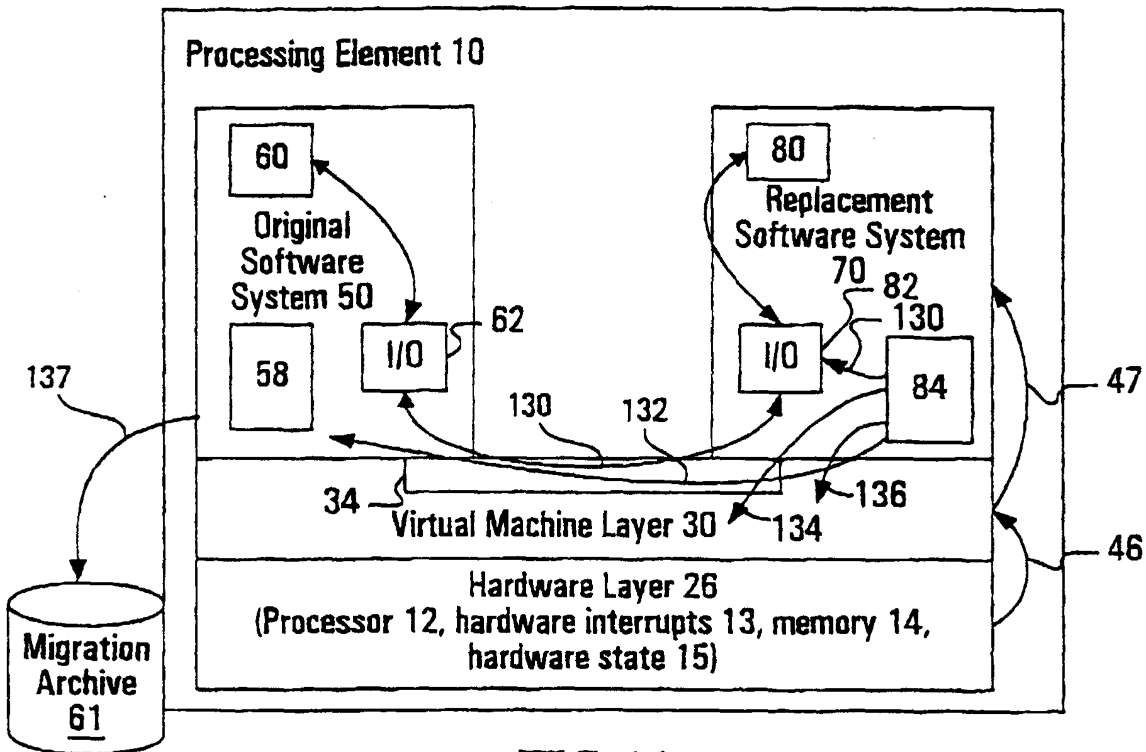

Processing Element 10
60
Original Software System 50
58
I/O
62
80
Replacement Software System 70
82
130
I/O
84
47
137
130
132
34
Virtual Machine Layer 30
134
136
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14, hardware state 15)
Migration Archive 61

FIG. 11

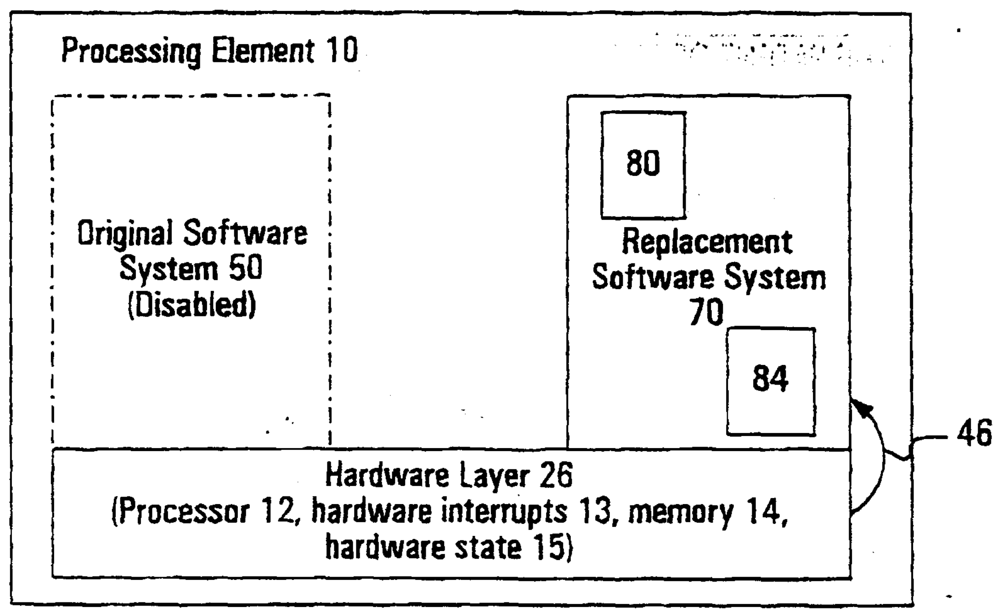

Processing Element 10
Original Software System 50 (Disabled)
80
Replacement Software System 70
84
46
Hardware Layer 26
(Processor 12, hardware interrupts 13, memory 14, hardware state 15)

FIG. 12

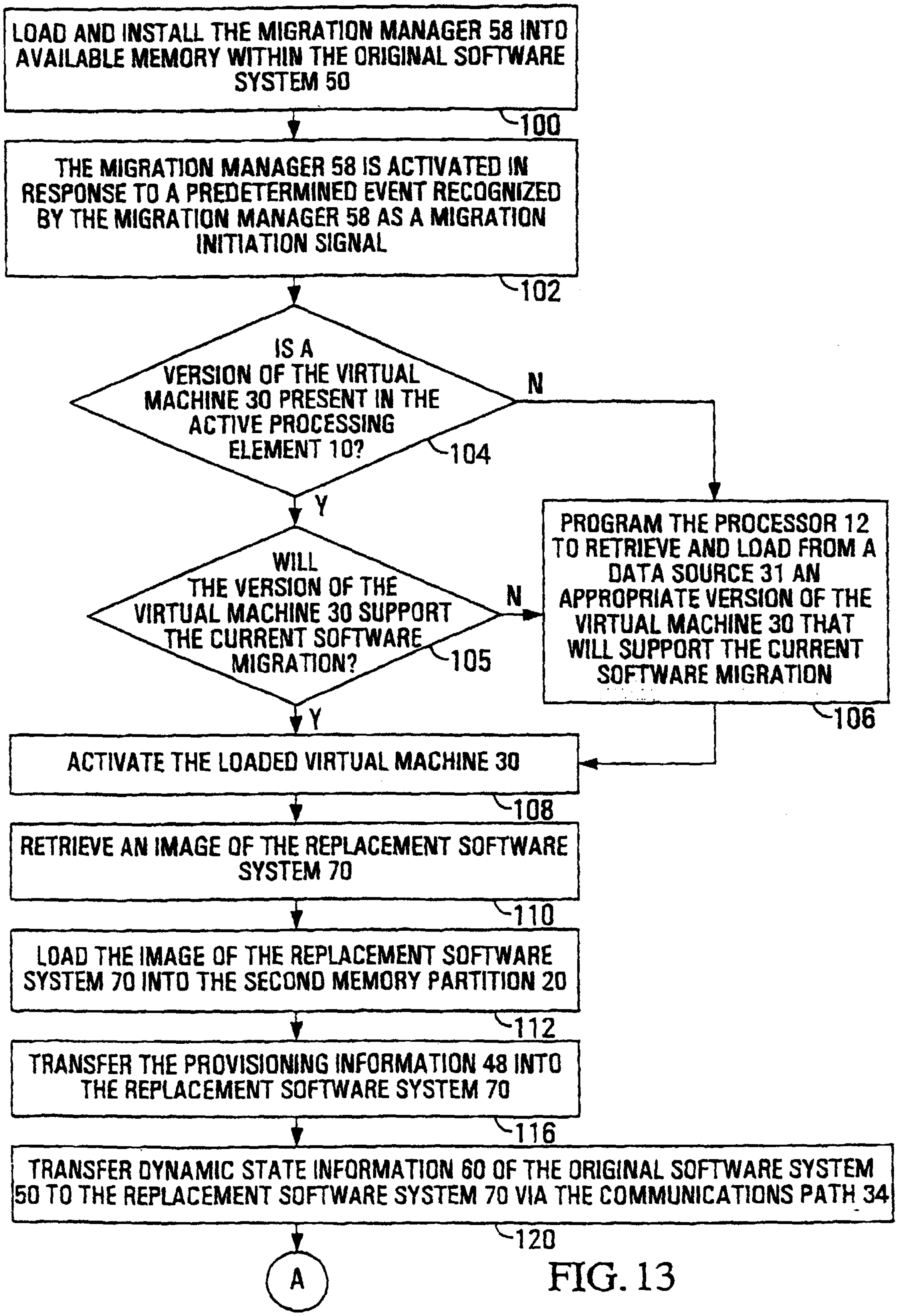
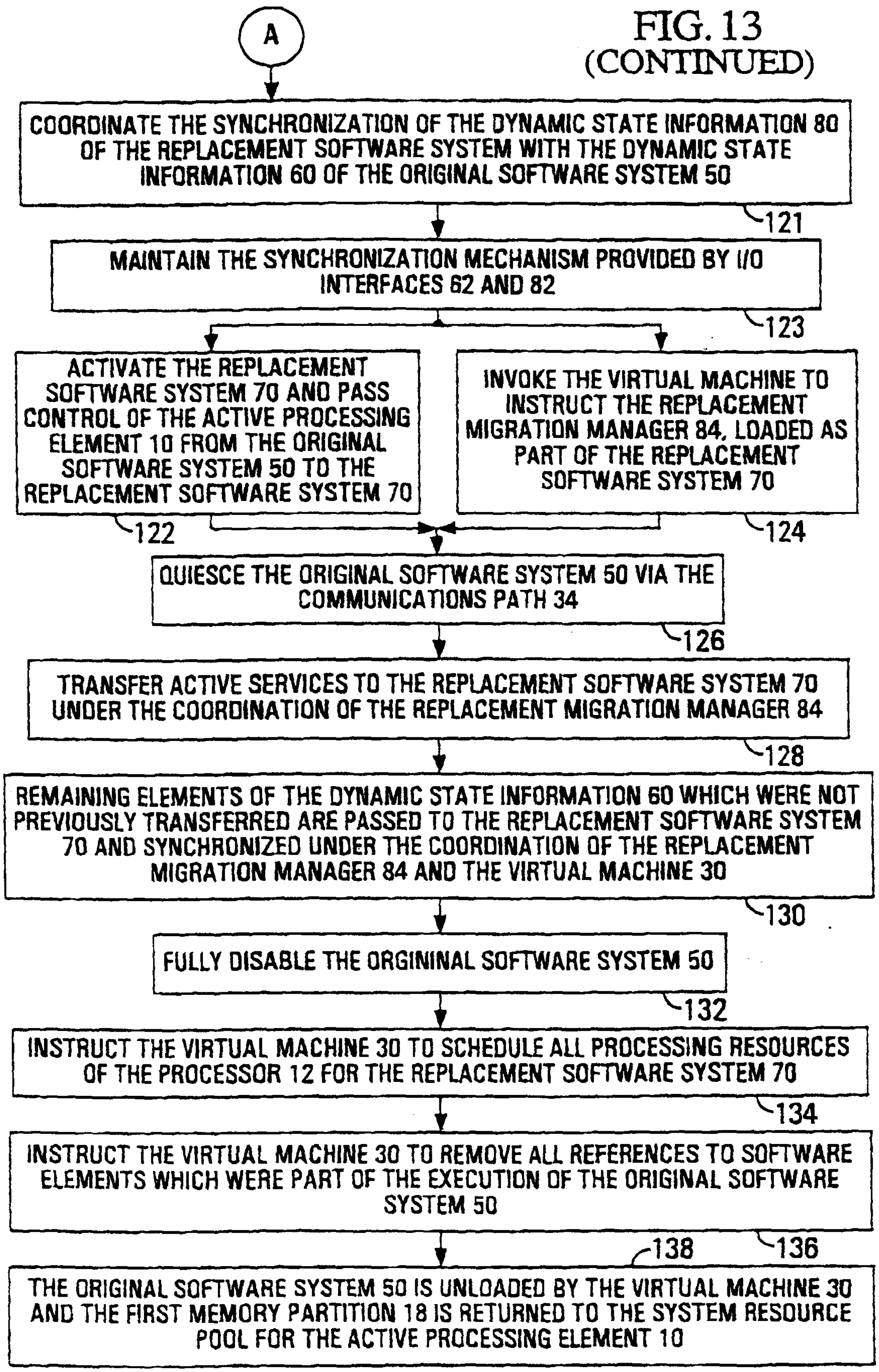

LOAD AND INSTALL THE MIGRATION MANAGER 58 INTO AVAILABLE MEMORY WITHIN THE ORIGINAL SOFTWARE SYSTEM 50
100
THE MIGRATION MANAGER 58 IS ACTIVATED IN RESPONSE TO A PREDETERMINED EVENT RECOGNIZED BY THE MIGRATION MANAGER 58 AS A MIGRATION INITIATION SIGNAL
102
IS A VERSION OF THE VIRTUAL MACHINE 30 PRESENT IN THE ACTIVE PROCESSING ELEMENT 10?
104
N
Y
WILL THE VERSION OF THE VIRTUAL MACHINE 30 SUPPORT THE CURRENT SOFTWARE MIGRATION?
105
N
Y
PROGRAM THE PROCESSOR 12 TO RETRIEVE AND LOAD FROM A DATA SOURCE 31 AN APPROPRIATE VERSION OF THE VIRTUAL MACHINE 30 THAT WILL SUPPORT THE CURRENT SOFTWARE MIGRATION
106
ACTIVATE THE LOADED VIRTUAL MACHINE 30
108
RETRIEVE AN IMAGE OF THE REPLACEMENT SOFTWARE SYSTEM 70
110
LOAD THE IMAGE OF THE REPLACEMENT SOFTWARE SYSTEM 70 INTO THE SECOND MEMORY PARTITION 20
112
TRANSFER THE PROVISIONING INFORMATION 48 INTO THE REPLACEMENT SOFTWARE SYSTEM 70
116
TRANSFER DYNAMIC STATE INFORMATION 60 OF THE ORIGINAL SOFTWARE SYSTEM 50 TO THE REPLACEMENT SOFTWARE SYSTEM 70 VIA THE COMMUNICATIONS PATH 34
120
A

FIG. 13

FIG. 13 (CONTINUED)



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 6038

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	WO 96 18146 A (ERICSSON TELEFON AB L M ;FUCHS ROBERT (SE); HOLTE ROST ANNA (SE);) 13 June 1996 (1996-06-13)	1,5-7	G06F9/445
Y	* page 14, line 27 - page 18, line 26 *	2,3	
A		8-26	
X	EP 0 809 181 A (SIEMENS AG) 26 November 1997 (1997-11-26)	17,21-25	
Y	* column 1, line 15 - column 4, line 45 *	18,19	
Y	EP 0 683 453 A (HITACHI LTD) 22 November 1995 (1995-11-22) * abstract *	2,18	
Y	US 5 359 730 A (MARRON ASSAF) 25 October 1994 (1994-10-25) * column 6, line 3 - column 7, line 40 *	3,19	
A	WO 96 10319 A (ERICSSON TELEFON AB L M) 4 April 1996 (1996-04-04) * page 8, line 6 - page 9, line 28 *	8-16	TECHNICAL FIELDS SEARCHED (Int.Cl.7)
A	EP 0 743 595 A (PHILIPS PATENTVERWALTUNG ;PHILIPS ELECTRONICS NV (NL)) 20 November 1996 (1996-11-20) * page 3, line 7 - page 4, line 2 *	1-26	G06F
A	WO 94 01819 A (ERICSSON TELEFON AB L M) 20 January 1994 (1994-01-20) * page 17, line 17 - page 29, line 25 *	1-26	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	10 November 2000	Bijn, K

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 00 30 6038

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

10-11-2000

Patent document cited in search report		Publication date	Patent family member(s)			Publication date
WO 9618146	A	13-06-1996	SE	504943	C	02-06-1997
			AU	717645	B	30-03-2000
			AU	4275796	A	26-06-1996
			BR	9509893	A	30-12-1997
			CA	2205481	A	13-06-1996
			CN	1169190	A	31-12-1997
			EP	0796461	A	24-09-1997
			FI	972405	A	06-06-1997
			JP	10510376	T	06-10-1998
			NO	972595	A	14-07-1997
			SE	9503339	A	10-06-1996
			US	6101327	A	08-08-2000
EP 0809181	A	26-11-1997	DE	59604300	D	02-03-2000
EP 0683453	A	22-11-1995	JP	7306844	A	21-11-1995
			CN	1123934	A,B	05-06-1996
			SG	30372	A	01-06-1996
			US	5729761	A	17-03-1998
US 5359730	A	25-10-1994	NONE			
WO 9610319	A	04-04-1996	US	5682533	A	28-10-1997
			AU	3623995	A	19-04-1996
			CN	1158662	A	03-09-1997
			EP	0784817	A	23-07-1997
			JP	10506763	T	30-06-1998
EP 0743595	A	20-11-1996	DE	19518266	A	21-11-1996
			JP	9034700	A	07-02-1997
			US	5987511	A	16-11-1999
WO 9401819	A	20-01-1994	US	5410703	A	25-04-1995
			AU	667559	B	28-03-1996
			AU	4516493	A	31-01-1994
			BR	9306651	A	08-12-1998
			CN	1081010	A,B	19-01-1994
			CN	1233011	A	27-10-1999
			DE	69326464	D	21-10-1999
			DE	69326464	T	03-02-2000
			DK	648353	T	03-04-2000
			EP	0648353	A	19-04-1995
			ES	2140462	T	01-03-2000
			FI	946195	A	30-12-1994
			GR	3031915	T	31-03-2000
			MX	9303648	A	31-01-1994

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 00 30 6038

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

10-11-2000

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9401819 A		NO 945096 A	22-02-1995
		US 5555418 A	10-09-1996

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82